Energy Monitoring System based on IoT

Serhii Tsyrulnyk[a,b], Volodymyr Tromsyuk[a], Maksym Tsyrulnyk[c] and Pavlo Rymar[c]

[a] *Vinnytsia Technical College, Khmelnytske highway, 91/2, Vinnytsia, 21021, Ukraine*
[b] *Vinnytsia National Agrarian University, str. Sonyachna, 3, City, Vinnytsia, 21008, Ukraine*
[c] *Vasyl' Stus Donetsk National University, str. 600-richchia, 21, Vinnytsia, 21021, Ukraine*

Abstract

Today, electrical appliances at home and work have become indispensable assistants that create comfortable conditions for people. Automated and intelligent electricity meters are being introduced around the world to monitor and assess electricity use in every home and workplace. The paper proposes a system designed for remote monitoring of electricity consumption from anywhere in the world and sends e-mail notifications when energy consumption reaches the threshold value. This technique uses a WeMos wireless control module with an ESP8266 microcontroller using a Wi-Fi channel, which is built into the Internet of Things and allows you to measure the current consumption of the load connected to the 220V mains and transmit the measured value via Wi-Fi. The ACS712 current sensor measures the current consumption and also provides insulation between the load and the IoT module. The Adafruit IO free cloud service and the MQTT Dashboard Android mobile app are used to track energy usage, and the IFTTT cloud service links the data to SMS / email notifications.

Keywords 1

Energy Monitoring System, Remote monitoring, Automatic billing, Internet of Things, module IoT

1. Introduction

The Internet of Things (IoT) is a rapidly growing industry, which includes technologies such as "Smart and Safe House", "Smart City", "Virtual Marketing", "Communication and Navigation Equipment", "Virtual Engineering". Such technologies already cover almost every segment in the industry, business, healthcare, and consumer goods. IoT is one of the main trends in the IT industry [1].

The Internet of Things (IoT) is a new field that is now developing rapidly. Internet-related things are designed to make life even more functional and comfortable. Thanks to intelligence and communications, a new set of functions appears in the equipment: monitoring, control, optimization, autonomy [2-4].

Connecting devices to the Internet provides new opportunities for both consumers and manufacturers. Consumers can control their costs and manage time, and manufacturers can control the operation of devices, more easily solve the problem of its maintenance, as well as offer new revenue-generating structures.

The main trend that affects the development of information systems in the energy sector is the Smart Grid concept. According to the Smart Grid concept, the priority direction of IT development in the energy sector in the coming years is the widespread introduction of smart (smart) meters. Smart electricity meters are provided with communication tools for the transmission of accumulated information using network technologies to monitor and make payments for utilities [5, 6].

ITTAP'2021: 1nd International Workshop on Information Technologies: Theoretical and Applied Problems, November 16–18, 2021, Ternopil, Ukraine
EMAIL: sovm@ukr.net (S. Tsyrulnyk); 2013tvd@gmail.com (V. Tromsyuk); maximtsyrulnyk@gmail.com (M. Tsyrulnyk); p.rymar@donnu.edu.ua (P. Rymar)
ORCID: 0000-0002-5703-9761 (S. Tsyrulnyk); 0000-0001-5022-8159 (V. Tromsyuk); 0000-0002-0647-2020 (P. Rymar)

2. System Architecture

The power monitoring system is a software and hardware module of wireless control using a Wi-Fi channel, which is built into the Internet of Things [2, 3, 7] and allows you to measure the current consumption of the load connected to the 220V mains and transmit the measured value via Wi-Fi network with remote access from a mobile phone, tablet or personal computer and sends an e-mail bill for electricity [8, 9].

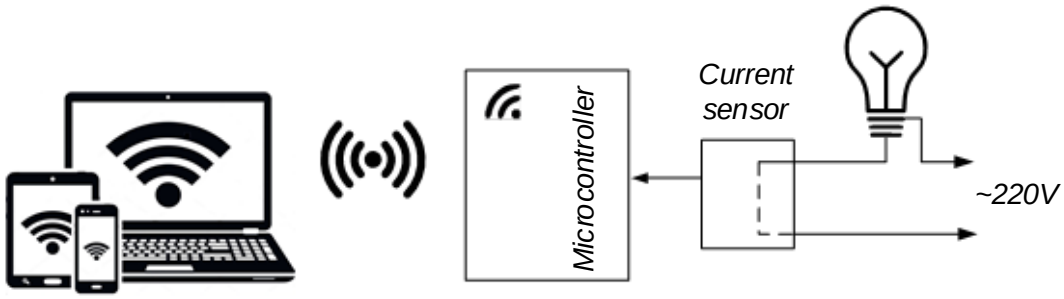

Figure 1: Block diagram of the electricity meter

2.1 Hardware

The main element of the electricity monitoring system is the IoT-based control module, which receives information from the current sensor and transmits via Wi-Fi to the cloud service, to which the user can connect and view information about electricity consumption.

The control module can be implemented in the following ways. The first way is to connect to an Arduino Uno and a Wi-Fi shield based on HDG04 [3, 11]. Hardware connection of these modules is performed by applying a Wi-Fi shield to the Arduino Uno, in the form of a "sandwich". Wi-Fi shield is based on the HDG104 module, which is a system on the chip, which provides an Arduino connected to the Internet via wireless LAN 802.11b / g (Wi-Fi). This assembly has strong disadvantages, namely: high price, large size.

The second way is to connect to the Arduino Pro Mini and ESP-01. In this connection, the Arduino Pro Mini acts as a microcontroller, and the ESP-01 (ESP8266) acts as an Internet transmission device. Since the level of the output signals of the microcontroller is 5V, there is no need for a level conversion device. Connections are made through a board to which two modules are soldered.

The third way is to use the IoT module on the ESP8266 chip in the design version of the Node MCU V3 or WeMos D1, which acts as a data transmission device via the Internet and a microcontroller control. This method has the most advantages in terms of characteristics, size, price and is therefore used to implement an electricity monitoring system.

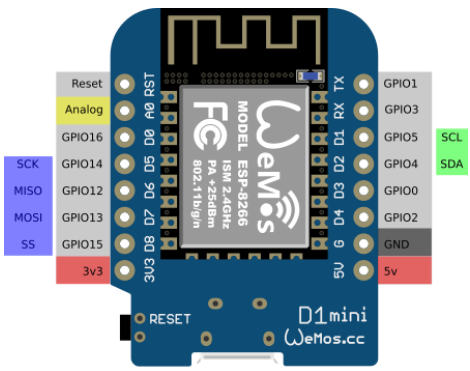

Figure 2: A pinout diagram of a WeMos D1 mini module and their corresponding functions

The port distribution of the WeMos D1 mini-module is shown in Figure 2. The module contains 11 general-purpose ports. Some ports have additional functions: GPIO1, GPIO3 - UART, GPIO5 / D1, GPIO4 / D2 - I²C / TWI, D5..D8 - SPI, D1..D7 - PWM outputs (PWM), A0 - analog output from ADC. A current sensor is connected to the module via A0.To connect this module, simply apply a supply voltage of + 5V.

A key component of the electricity monitoring system is the current sensor ACS712 [10] from Allegro. The ACS712 module works on the principle of the Hall effect: if a conductor with a current is placed in a magnetic field, then at its edges there is an EMF directed perpendicular to both the direction of the current and the direction of the magnetic field. The Hall sensor is used to measure the magnetic field around a current-carrying conductor. The result of the measurement is called the Hall voltage. This Hall voltage is proportional to the current flowing through the conductor. The main advantage of using the ACS712 current sensor is that it can measure both AC and DC, and also provides isolation between the load (AC / DC load) and the measuring unit (part of the microcontroller). Technical characteristics of the ACS712-30A sensor [10]: sensitivity 185 mV / A, supply voltage +5.0 V; current consumption does not exceed 11 mA; current bus resistance 1.2 mOhm. The diagram of the ACS712-30A module is shown in Figure 3.

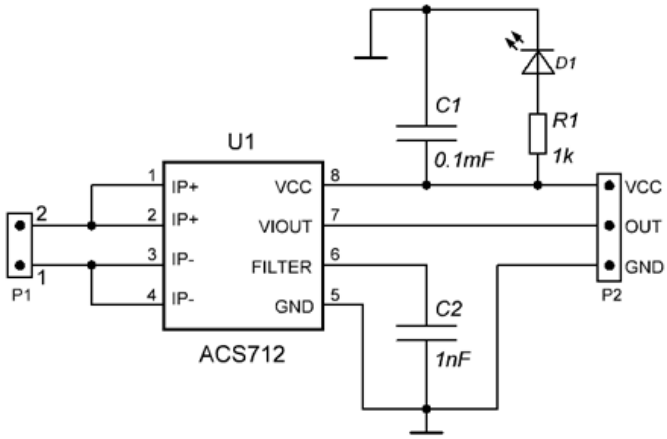

Figure 3: Diagram of the ACS712 module

As shown in Figure 3, connector P2 is used to connect to a microcontroller (WeMos). Connector P2 is designed to pass current through it. The module operates at + 5 V, so Vcc must be powered from 5 V, and the ground must be connected to the ground of the system. Output Vout has a bias voltage of 2500 mV. When there is no current through the wire, then the output voltage will be 2500 mV, and when the current is positive, the voltage will be more than 2500 mV, and when the current is negative, the voltage will be less than 2500 mV.

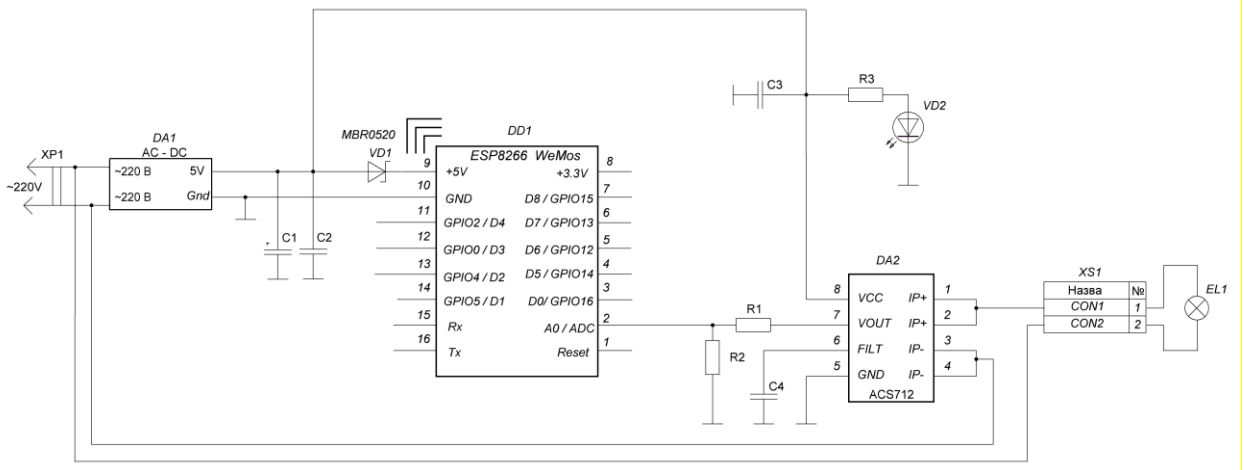

Figure 4: Connecting the ACS712 current sensor to the microcontroller

The analog output A0 of the WeMos module will be used to read the output voltage (Vout) of the module, which will be 512 (2500 mV) when the current does not pass through the wire. A resistive divider will be used to match the levels between the WeMos module and the ACS712. In Figure 4 shows that to measure current, the sensor is connected to the circuit between the power supply and the load. Table 1 helps to understand how the output voltage and ADC value change depending on the current flowing through the wire.

Table 1
The value of the output voltage and ADC depending on the current

Analog Value	Vout (mV)	Current Through the Wires (A)
1023	5000	13.51351351
800	3910.068426	7.621991493
700	3421.309873	4.980053367
512	**2502.443793**	**0.013209691**
300	1466.27566	-5.587699136
301	1471.163245	-5.561279755
0	0	-13.51351351

The values in Table 1 were calculated based on the information provided in the datasheet ACS712-30A [10]. They can be calculated by the following formulas using the following formulas:

$$Vout\ (mV) = (ADC\ value / 1023) * 5000$$

$$Current\ (A) = (Vout\ (mV) - 2500) / 185.$$

An AC-DC Hi-Link HLK-PM01 voltage converter with a 5V output voltage and a maximum output current of 0.6A is used to power the WeMos D1 mini-module and the ASC712 sensor with a voltage of + 5V from the mains voltage. The unit has a small sealed housing with legs for mounting on a printed circuit board. There is built-in protection against short circuits.

2.2. Software

2.2.1. Web-integration

Various sensors transmit information about different physical quantities through the appropriate channels, while consumers subscribe to receive them. MQTT is the most popular Internet of Things protocol. MQTT is a subscription publication protocol that facilitates one-to-many communication. Clients can post messages to the broker and/or subscribe to the broker to receive certain messages. Messages are organized by topics that act as a system for sending messages to subscribers [4].

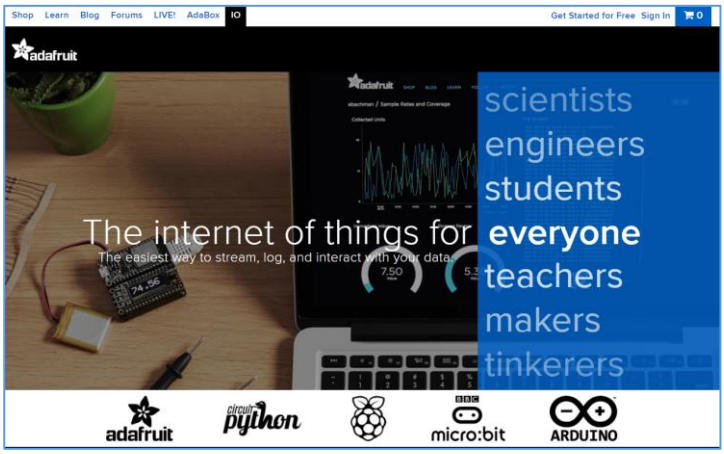

Figure 5: Start page of the Adafruit IO service

An MQTT broker must be used to monitor energy consumption over the Internet. The Adafruit IO platform is used as an MQTT broker [1, 2, 9]. The procedure for setting up the service is given below.

In Figure 5 shows the login or registration for the Adafruit IO service with your credentials. Go to the IO tab at the top of the Adafruit IO service page which is shown in Figure 6.

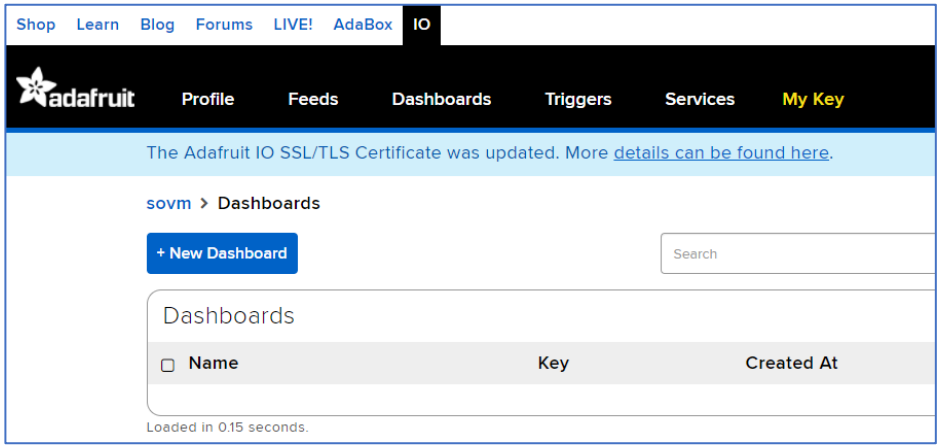

Figure 6: IO tab Adafruit IO service page

In Figure 7 shows how to create a new information panel (Dashboard).

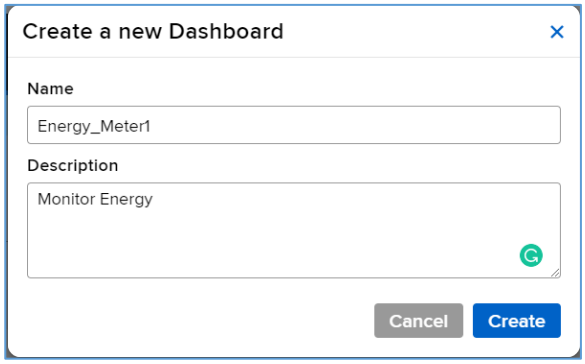

Figure 7: Creating an information panel

Click the My Key button and write down the value of Active Key, which will be used in the program code, as shown in Figure 8.

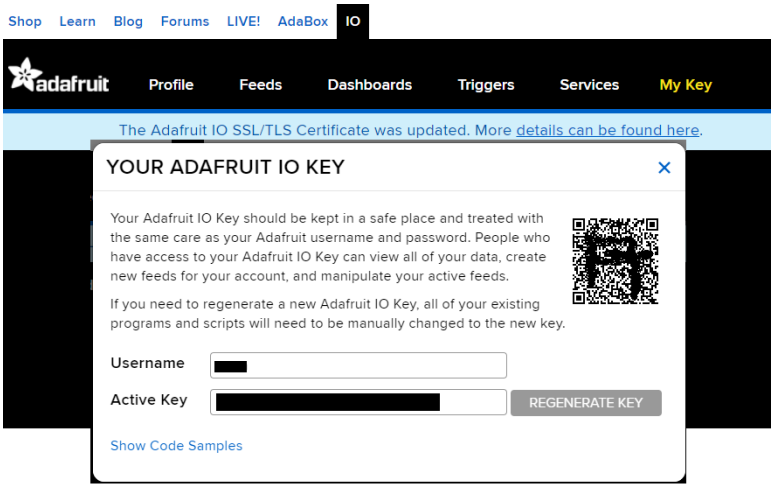

Figure 8: Viewing the value of the Adafruit IO Key project

In the Dashboard properties, select the Create New Block command. Click on the "Gauge" button to display the energy usage level, as shown in Figure 9.

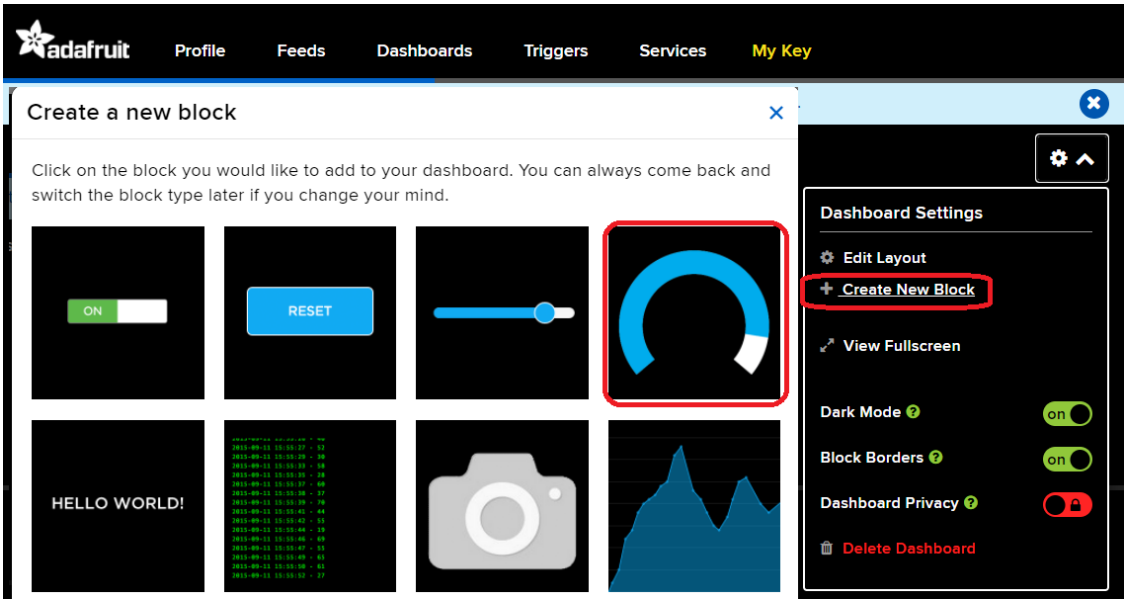

Figure 9: Selecting the Gauge block for the information panel

In the settings of the "Sensor" block, fill in the name (Power) and the minimum and maximum values from 0 and 100, respectively, as shown in Figure 10.

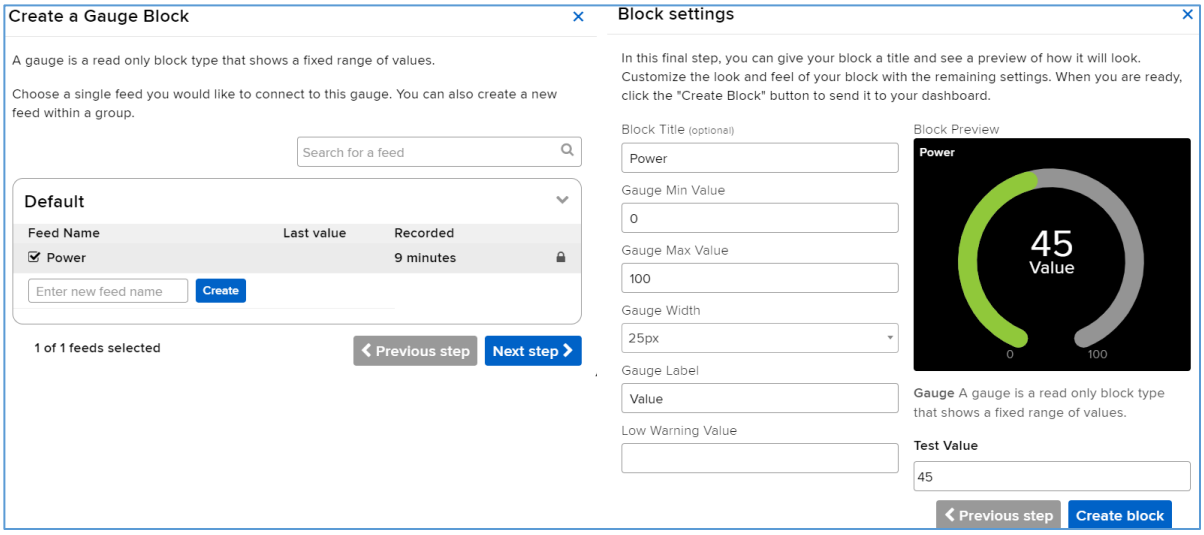

Figure 10: Configuring the properties of the "Sensor" block

On the Dashboard, we add the block of type "Text" and we adjust similarly its properties, as shown in Figure 11. As a result, the information panel will look like Figure 12.

Automation connects the Internet of Things web services. IFTTT [2, 12] is a leading application for automation. Using a series of conditional statements, you can force certain events to trigger certain actions. The IFTTT service has a very low entry threshold, free, simple, integrates with a huge number of services, so it is advisable to use it for integration with the Adafruit IO platform, which will allow you to send the results of electricity monitoring by e-mail. Consider the settings of the IFTTT service. Figure 13, 14 shows how to configure the Adafruit service to work with the IFTTT service [9, 12].

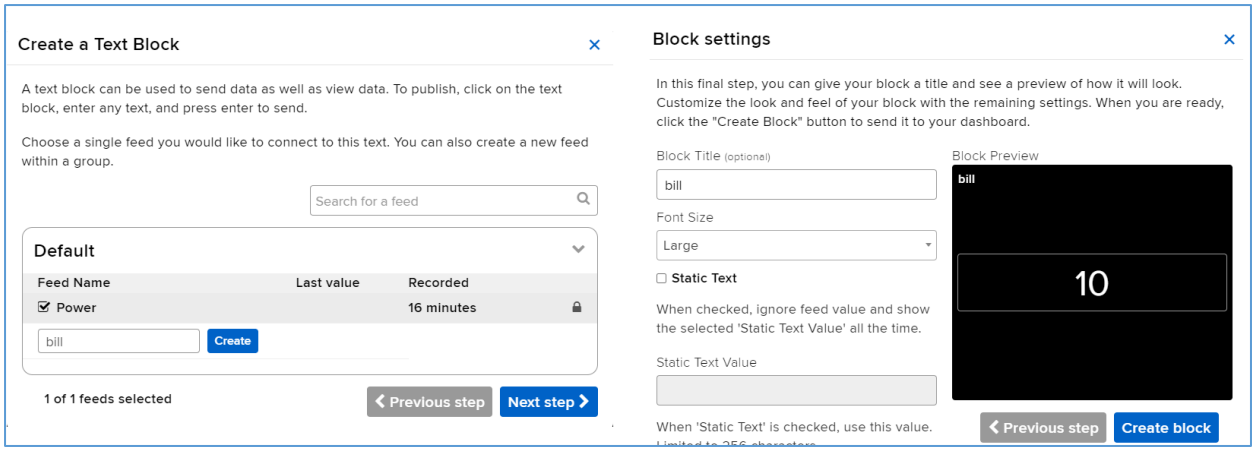

Figure 11: Configuring the properties of the "Text" block

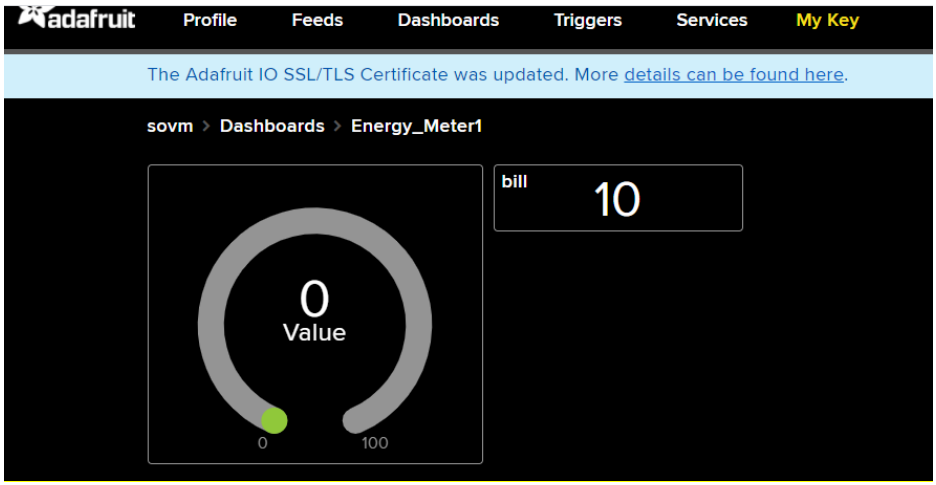

Figure 12: Appearance of the energy meter information panel

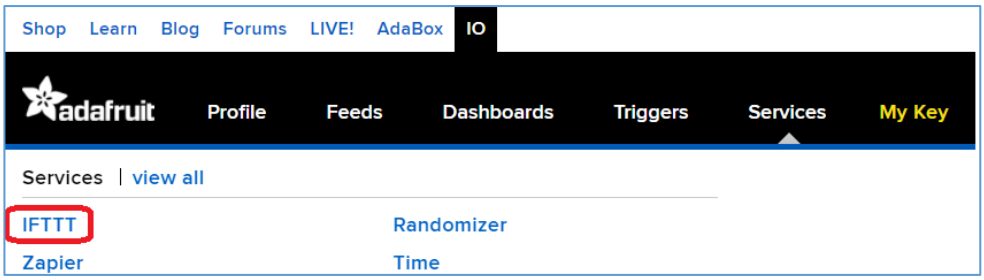

Figure 13: Selecting the IFTTT service to work with the Adafruit service

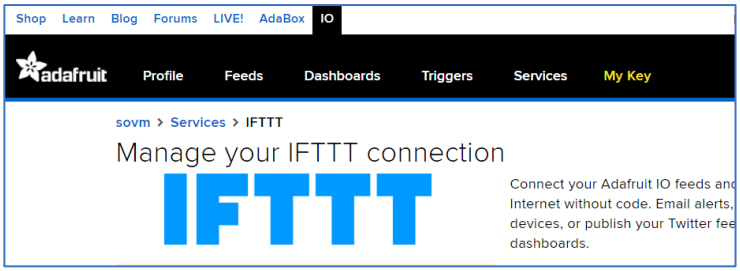

Figure 14: Configuring the Adafruit service to work with the IFTTT service

Next, create an applet in IFTTT [2, 12] to send an e-mail to the energy meter, if the value of the

electricity bill is equal to or exceeds a certain value. To do this, go to the IFTTT service from your account and select the Create command, as shown in Figure 15.

In the window for creating a new applet, see Figure 16, select the Add command to select the service. In Figure 17, in the search window, enter Adafruit and select the applet "Monitor a feed on Adafruit IO", which is shown in Figure 18.

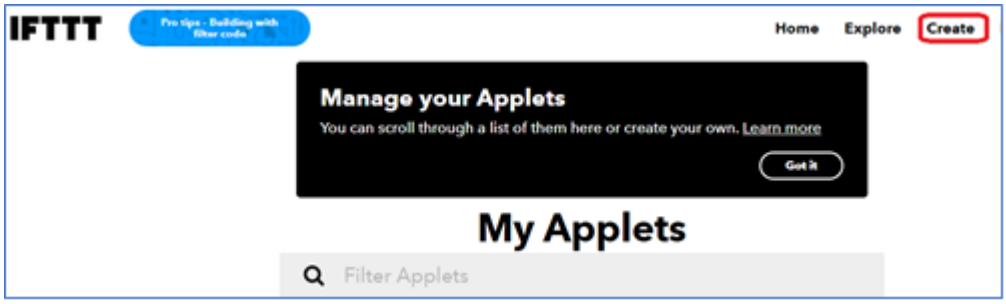

Figure 15: Creating a new applet in the IFTTT service

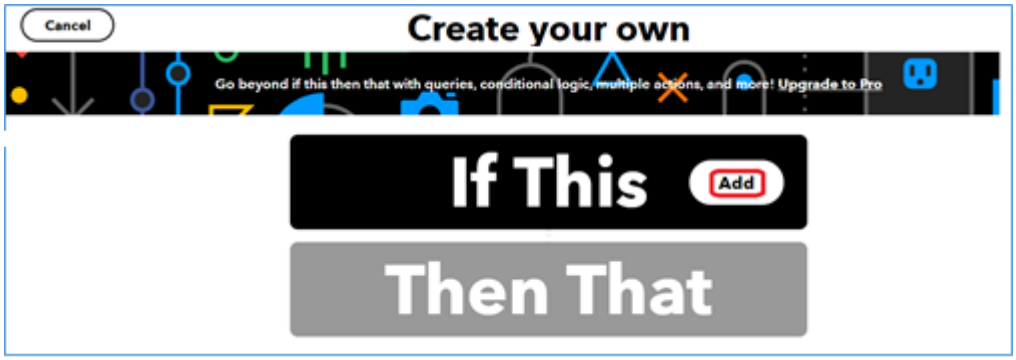

Figure 16: Selecting a service to create a new event

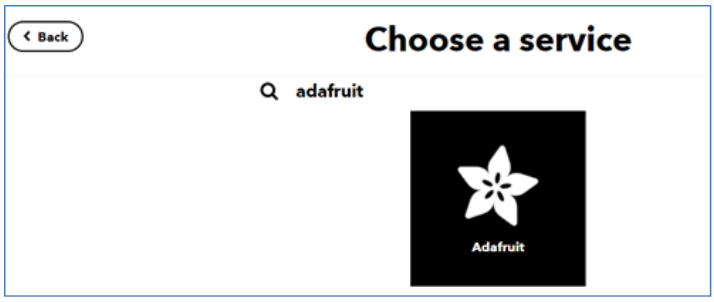

Figure 17: Searching for the Adafruit event source service

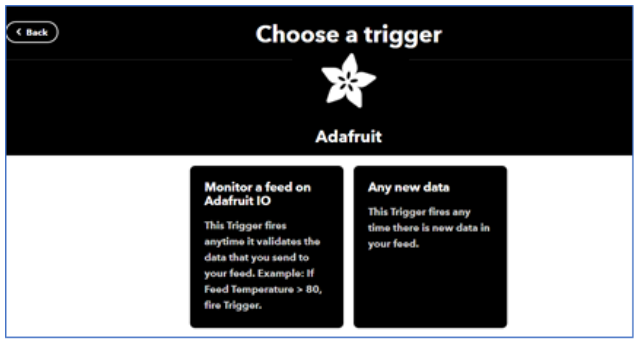

Figure 18: Adafruit event source applets

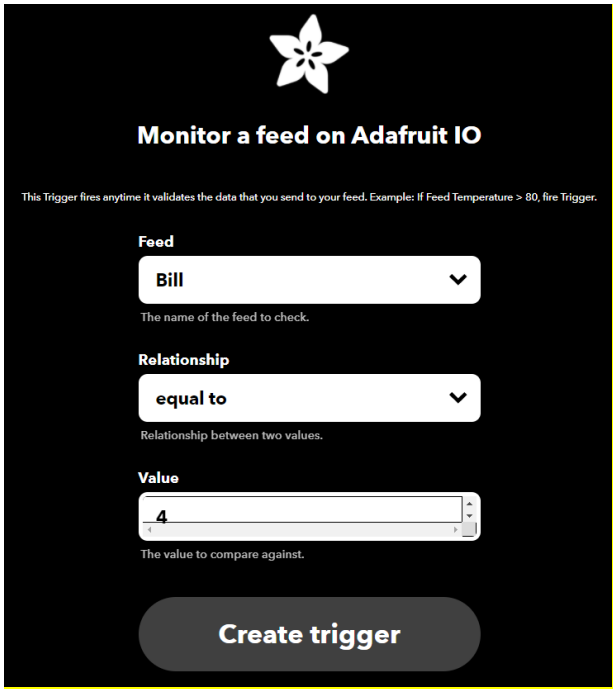

Figure 19: Configuring the Monitor a feed on Adafruit IO applet

Figure 20 shows how to configure the fields of the Monitor a feed on the Adafruit IO applet to work with the Adafruit dashboard. After clicking the "Create trigger" button, a window for selecting an action on the event from the Adafruit applet opens. You need to select the Add button and go to search for the Gmail service, select its applets, and configure the applet "Send yourself an email", which is shown in Figures 21, 22, 23.

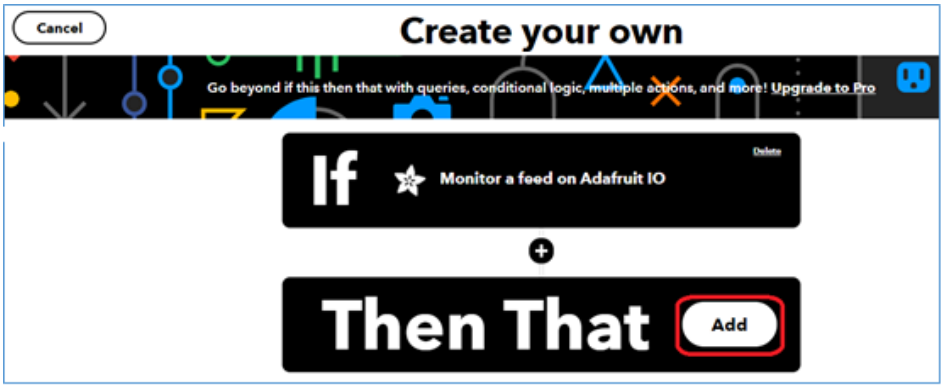

Figure 20: Adding an action from the Adafruit applet

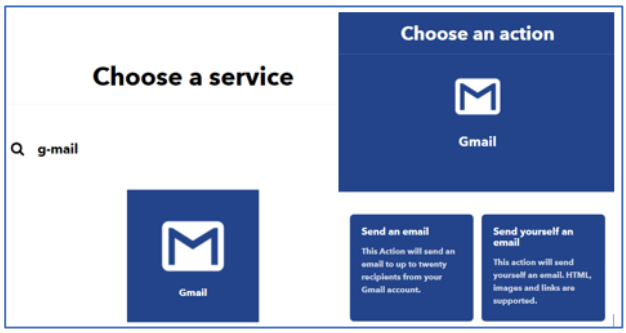

Figure 21: Select a Gmail service to send an email to an event

After clicking the "Create action" button, the IFTTT service window opens, which is all configured: when an event occurs in Adafruit, an email will be sent to the user's e-mail, which is shown in Figures 22, 23.

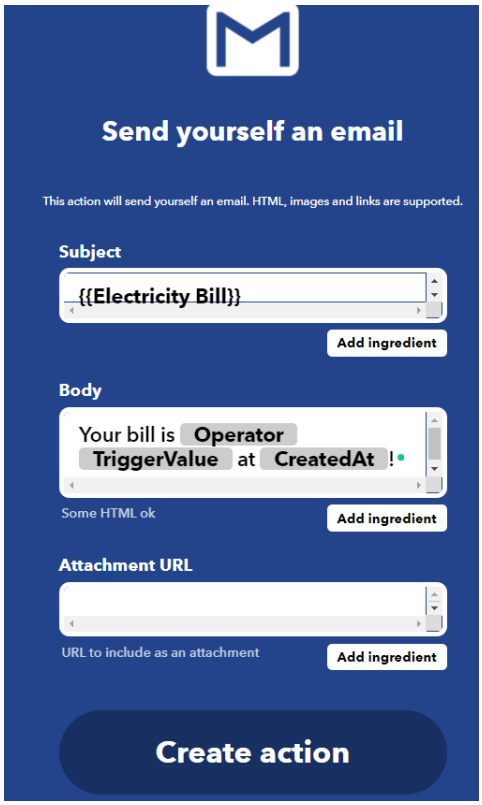

Figure 22: Setting up the applet “Send yourself an email”

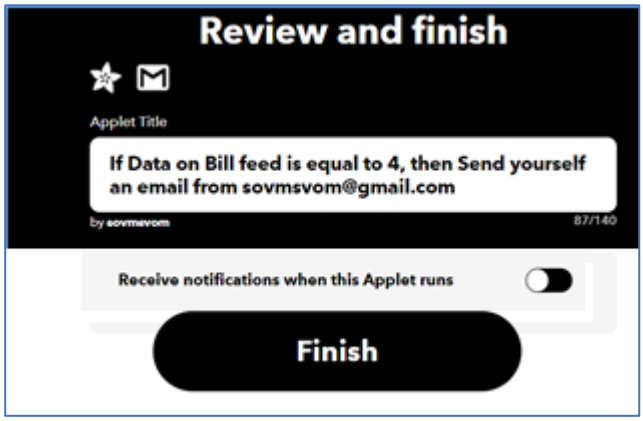

Figure 23: Completing the Adafruit and Gmail interaction setup

2.2.2. IoT module software

The software for the ESP8266 microcontroller was developed in the Arduino IDE environment [2, 4, 13].

The ACS712.h library is used to operate the ACS712 sensor [10].

The ACS712.h library has a built-in function for calculating current. The command is used to connect the library

```
#include "ACS712.h"
```

Create an array to store energy to send it to the NodeMCU.

```
char watt [5];
```

Create an instance to use ACS712-30Amp, which is connected to A0.

```
ACS712 sensor(ACS712_30A, A0);
```

In the *setup* function, we determine the data rate 115200 with WeMos and call the *sensor.calibrate()* function to calibrate the current sensor to obtain accurate measurement readings.

```
void setup () {
  Serial.begin (115200);
  sensor.calibrate ();
}
```

In the *loop* function, call the *sensor.getCurrentAC()* function to obtain the current value and save it in variable *I*. After receiving the current, calculate the power by the formula $P = V * I$.

```
void loop () {
  float V = 220;
  float I = sensor.getCurrentAC ();
  float P = V * I;
```

Next, the code converts the power to $W * h$ *(Wh)*.

```
  last_time = current_time;
  current_time = millis();
  Wh = Wh + P *((current_time -last_time)/3600000.0);
```

Convert the value of Wh to characters to send it from WeMos, using the function *dtostrf ()*; will convert float to a char array so that it can then be easily printed:

```
dtostrf(floatvar, StringLengthIncDecimalPoint, numVarsAfterDecimal, charbuf);
```

The AdaFruit MQTT library is required to work with WeMos. We include all libraries for the Wi-Fi ESP12 module and AdaFruit MQTT.

```
#include <ESP8266WiFi.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"
#define WLAN_SSID "xxxxxxxx"; SSID/назва Wi-Fi
#define WLAN_PASS "xxxxxxxxxxx"; пароль Wi-Fi
```

Next, configure the connection to the AdaFruit service (server, port, username and code (Figire 2.8))

```
#define AIO_SERVER "io.adafruit.com"
#define AIO_SERVERPORT 1883
#define AIO_USERNAME "********";ім’я користувача AdaFruit
#define AIO_KEY "**********************";код проекту AdaFruit
```

Create a WiFiClient class to connect to the MQTT server, configure the MQTT client class by passing the WiFi client and the MQTT server and login data.

```
WiFiClient client;
Adafruit_MQTT_Client mqtt(&client, AIO_SERVER, AIO_SERVERPORT, AIO_USERNAME,
AIO_KEY);
Adafruit_MQTT_Publish Power = Adafruit_MQTT_Publish(&mqtt, AIO_USERNAME
"/feeds/Power");
Adafruit_MQTT_Publish bill = Adafruit_MQTT_Publish(&mqtt, AIO_USERNAME "/feeds/bill");
```

In the *setup ()* function, add the information output setting to the Port Monitor and connect the Wi-Fi module to the Wi-Fi access point.

```
void setup() {
  Serial.begin(115200);
  delay(10);
  Serial.println(F("Adafruit MQTT demo"));
  // Connect to WiFi access point.
  Serial.println(); Serial.println();
```

```
  Serial.print("Connecting to ");
  Serial.println(WLAN_SSID);
  WiFi.begin(WLAN_SSID, WLAN_PASS);
 ....
 }
```

In the loop function we publish data in AdaFruit IO.

```
void loop() {
  MQTT_connect();
  int i=0;
  float watt1;
```

The next function calculates the amount of the bill by multiplying the power (in W · h) by the energy tariff and dividing it by 1000 to make the power in kWh.

```
bill_amount = watt1 * (energyTariff/1000); // 1unit = 1kwH
```

The data can then be published to the MQTT broker

```
Serial.print(F("\nSending Power val "));
Serial.println(watt1);
Serial.print("...");
if (! Power.publish (watt1)) {Serial.println (F ("Failed"));}
else {Serial.println (F ("Добре!"));}
if (! bill.publish (bill_amount)) {Serial.println (F ("Failed"));}
else {Serial.println (F ("Добре!")); }
```

2.3. Problem Faced

In this system, the WeMos module and the ACS712 current sensor are very important elements on which the accuracy of measuring the consumed electricity depends. The ACS712.h library provides the ability to calibrate the ACS712 sensor programmatically. Therefore, the electricity monitoring system needs to be calibrated before operation.

3. ACS712 current sensor operation and calibration test

A computer model of the power monitoring system in the Proteus 8.7 software environment was developed to study the operation and calibration of the ACS712 current sensor, which is shown in Figure 24 [14].

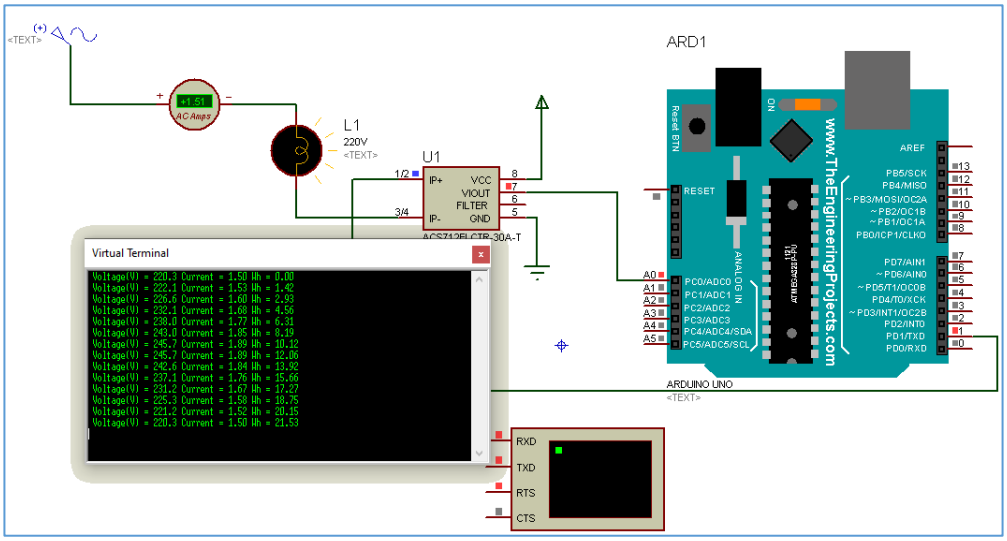

Figure 24: Scheme of modeling of an electricity monitoring system based on IoT

Figure 24 Virtual Terminal simulates the operation of the WiFi module and allows you to check the data that the monitoring system transmits to the cloud service or to the mobile application. A test program was recorded for the Arduino module, the purpose of which was to check the operation of the ACS712 sensor, the reliability of the data (voltage, current, energy) that the Arduino transmits to the Virtual Terminal. Listing test program 1:

```
const int currentPin = A0;
int sensitivity = 66;
int adcValue= 0;
int offsetVoltage = 121;
double adcVoltage = 0;
double currentValue = 0;
double p;
double  Wh =0 ;
double bill_amount = 0;
unsigned int energyTariff = 1.68;
unsigned long last_time =0;
unsigned long current_time =0;

void setup() {
  Serial.begin(9600);
}

void loop(){
  adcValue = analogRead(currentPin);
  adcVoltage = (adcValue / 1024.0) * 466;
  currentValue = ((adcVoltage - offsetVoltage) / sensitivity);
  Serial.print("\t Voltage(V) = ");
  Serial.print(adcVoltage,1);
  Serial.print("\t Current = ");
  Serial.print(currentValue,2);
  p = adcVoltage * currentValue;
  last_time = current_time;
  current_time = millis();
  Wh = Wh + p *(( current_time -last_time) /3600000.0) ;
  Serial.print("\t Wh = ");
  Serial.println(Wh,2);
  delay(10000);
  bill_amount = Wh * (energyTariff/1000); // 1unit = 1kwH
  if (bill_amount > 4){
    for (int i =0; i<=2; i++){
       Serial.print("\t Bill_amount = ");
       Serial.println(bill_amount);
       delay(5000);}
  }
  delay (5000);
}
```

Figure 24 uses an AC generator (220V / 50Hz), a lamp (220V / 40W), and an AC ammeter. Every 10 seconds, data on voltage, current, consumed electricity, electricity bills are transmitted to the Virtual Terminal if the consumed electricity exceeds 4 kW. The results of modeling the IoT-based monitoring system are shown in Figure 25.

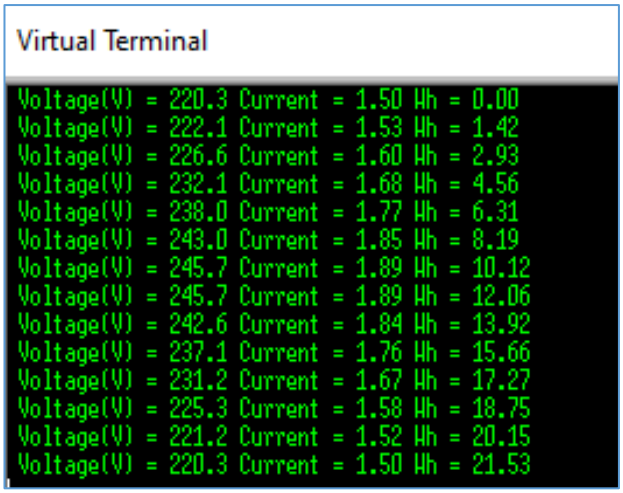

Figure 25: Results of modeling the electricity monitoring system

The obtained data correspond to the readings of the ammeter, which is connected in series with the load and the value of the mains voltage, however, the real scheme will need to adjust the value issued by the sensor ACS712. The manufacturer recommends setting the sensitivity value for the ACS712-30A sensor to 66 mV / A. We will make changes to the test program so that the user can change the values of sensitivity and FormFactor. To do this, add another Virtual Terminal to the scheme, which will simulate commands coming from a mobile application or cloud service. The test program provides commands "*" or "/", which increase or decrease by 1 the value of the sensor sensitivity, and "+", "-", which increase/decrease the FormFactor by 1.05. Test program to study the effect of sensitivity and FormFactor values on measurement accuracy:

```
#include "ACS712.h"
// Arduino UNO has 5.0 volt with a max ADC value of 1023 steps
// ACS712 5A uses 185 mV per A, 20A-100 mV, 30A-66 mV
ACS712  ACS(A0, 5.0, 1023, 66);

void setup(){
  Serial.begin(9600);
  ACS.autoMidPoint();
}
void loop(){
  int mA = ACS.mA_AC();
  Serial.println(mA);
  if (Serial.available() > 0)
  {
    char c = Serial.read();
    if (c == '*') ACS.setmVperAmp(ACS.getmVperAmp() + 1);
    if (c == '/') ACS.setmVperAmp(ACS.getmVperAmp() - 1);
    Serial.print("mVperAmp:\t");
    Serial.println(ACS.getmVperAmp());
    if (c == '+') ACS.setFormFactor(ACS.getFormFactor() * 1.05);
    if (c == '-') ACS.setFormFactor(ACS.getFormFactor() / 1.05);
    Serial.print("formFactor:\t");
    Serial.println(ACS.getFormFactor());
  }
  delay(1000);
}
```

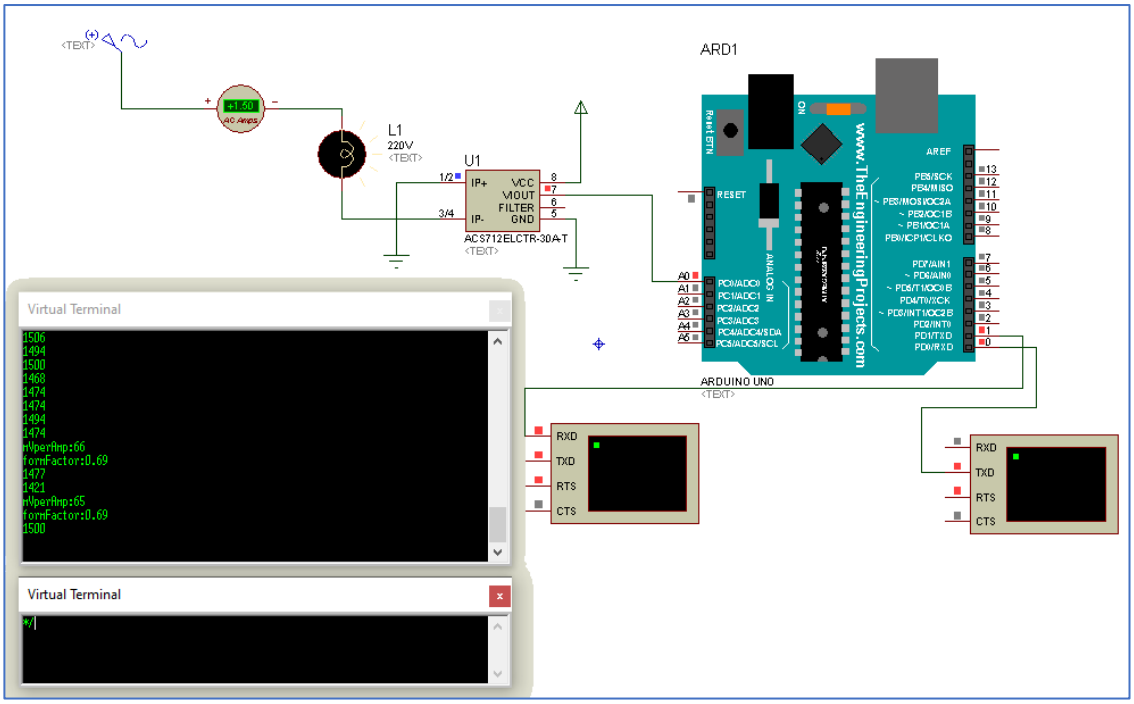

Figure 26: Investigation of the influence of sensitivity and FormFactor values on the measurement accuracy of the ACS712-30 sensor.

The results of the study of the influence of sensitivity and FormFactor values on the measurement accuracy of the ACS712-30 sensor are shown in Table 2 and Figures 26, 27.

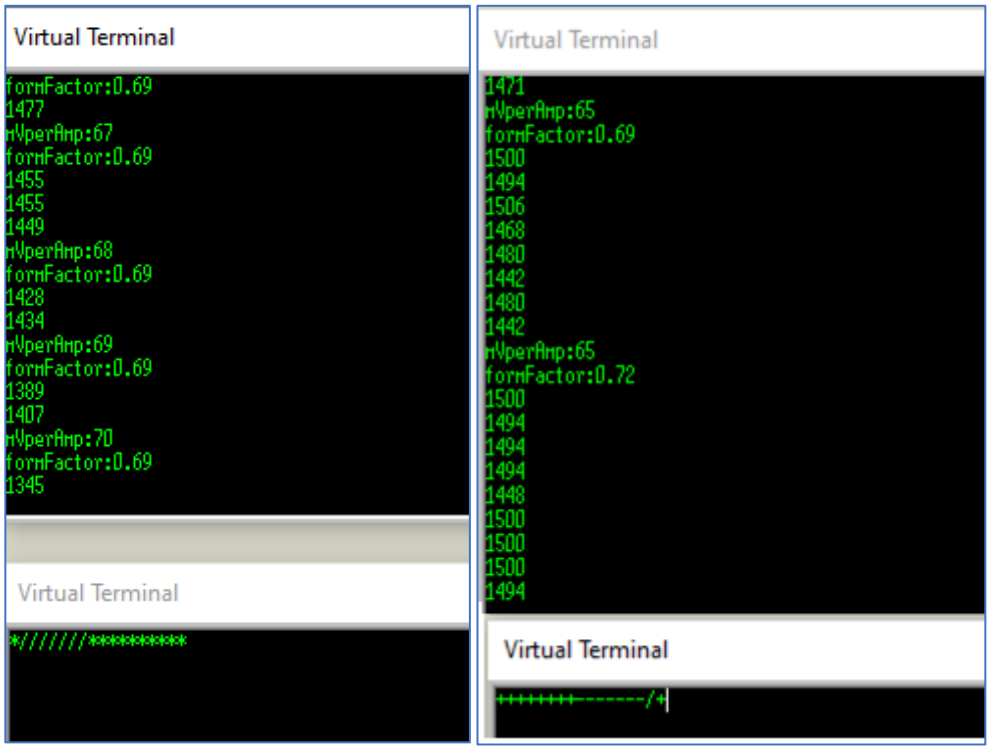

Figure 27: Investigation of the influence of sensitivity and FormFactor values on the measurement accuracy of the ACS712-30 sensor

According to the results of the study, it was found that during the simulation, the most accurate values of the current measurement of the sensor ACS712-30 are achieved at the value sensitivity = 65

and FormFactor = 0.72.

Table 2
Study of the influence of sensitivity values on the measurement accuracy of the sensor ACS712-30

mVperAmp	60	61	62	63	64	65
I_ASC712, mA	1632	1592	1566	1521	1517	1500
mVperAmp	66	67	68	69	70	
I_ASC712, mA	1471	1455	1428	1389	1345	

4. Results

The power monitoring system based on the IoT module is based on the ESP8266 microcontroller, which is part of the WeMos D1 mini platform. The microcontroller organizes data transfer via WiFi to the MQTT broker. ACS712 is used as a current sensor, which can measure both alternating and direct current, and also provides insulation between the load (alternating/direct current load) and the measuring unit (part of the microcontroller). A prototype of the electricity monitoring system is shown in Figure 28.

Figure 28: Appearance of an IoT-based electricity meter

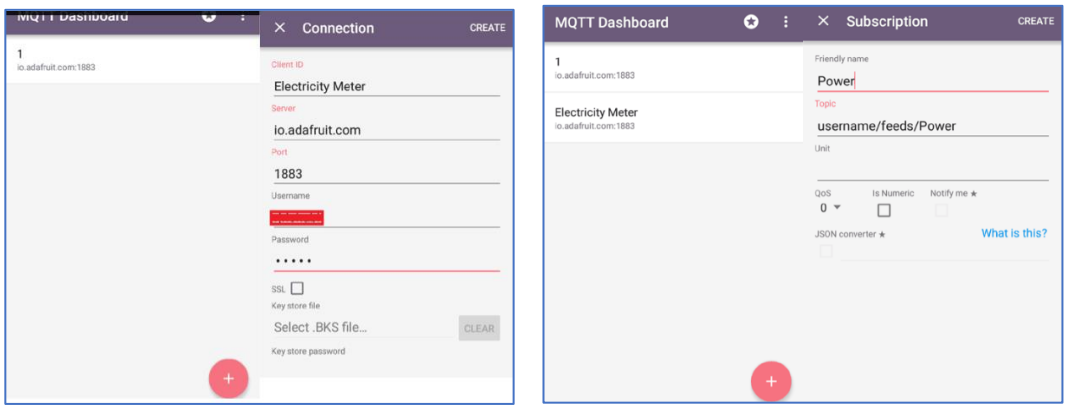

Figure 29: MQTT Dashboard mobile application configuration window for IoT-based power monitoring system

Launch the MQTT Dashboard application on the mobile phone and configure it according to Figure 29. Figure 30 shows the window of the mobile application, which allows you to monitor the values of Power and Bill.

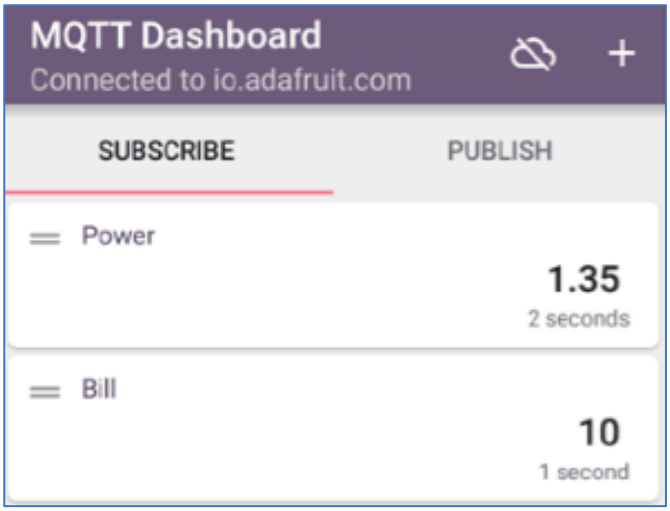

Figure 30: MQTT Dashboard mobile application Window

Check the values in the Power field of the application on the mobile phone in half an hour. If the value exceeds or is less than 37.5, ACS712-30 must be calibrated. To do this, enter a sensitivity factor in the program code (line 61) and reprogram the WeMos module.

5. Conclusion

The Internet of Things captures virtually every segment of the industry, business, healthcare, and consumer goods. Smart devices for the home such as irrigation systems, code access, and protection, lighting, thermostats are easily implemented through the IFTTT service, which allows you to combine a variety of network applications, as well as the system "Smart Home".

Going to the Internet of Things is quite simple. You must select a hardware platform, development environment, and create a graphical interface for your device.

For Internet of Things projects, the question arises of choosing a mobile application or online service. For non-commercial projects, the most optimal connection is the AdaFruit IO online platform, the IFTTT service, and the MQTT Dashboard application for the Android platform. They will allow you to access private or public online access from any device that has access to the Internet and the ability to have its mobile application interface. This connection allows you to monitor the data.

The article proposes practical aspects of the implementation of the electricity monitoring system based on the IoT module WeMos and the sensor ACS712-30 with a minimum cost and the ability to adapt to the home automation system. The use of wireless technologies provides several advantages: ease of installation and operation, scalability and ease of solution, integration of mobile devices.

The proposed system allows you to quickly monitor and analyze the mode of consumption of electricity and power by major consumers of electrical equipment over long distances, which increases comfort. Wireless technology is an attractive choice when renovating, repairing, and installing new systems.

6. References

[1] V. Kharchenko (Ed.), Internet of Things for Industry and Human Application, volume 1 of Fundamentals and Technologies, Ministry of Education and Science of Ukraine, National Aerospace University KhAI, 2019 (in English).

[2] M. Schwartz, Internet of Things with ESP8266, 2nd. ed., Saint Petersburg, BHV-Peterburg, 2019 (in Russian).
[3] V. Petin, Arduino and Raspberry Pi in the Internet of Things projects, 2nd. ed., Saint Petersburg, BHV-Peterburg, 2019 (in Russian).
[4] S. Tsyrulnyk, Mobile applications and online WI-FI monitoring platforms of weather stations, Open educational e-environment of modern University, volume 9, 2020, pp. 181–192. doi: 10.28925/2414-0325.2020.9.15. (in Ukrainian).
[5] R. Subhash, V. Balaji, K. Sanjana, N. Madhavan and T. Siva, Smart Automated Energy Meter, Communications in Computer and Information Science, vol 1214, 2020. doi: 10.1007/978-981-15-7219-7_2 (in English).
[6] Smart Grid. Umnyie Seti. Intellektualnyie seti elektrosnabzheniya, 2020. URL: https://lnnk.in/dece (in Russian).
[7] A. Belov, Control of the ARDUINO module via Wi-Fi from mobile devices, 1nd. ed., Saint Petersburg, Science and Technology, 2020 (in Russian).
[8] IoT Based Smart Energy Meter, 2021. URL: https://iotdesignpro.com/projects/iot-based-smart-energy-meter (in English).
[9] IoT Based Electricity Energy Meter using ESP12 and Arduino, 2018. URL: https://circuitdigest.com/microcontroller-projects/iot-electricity-energy-meter-using-esp12-arduino. (in English).
[10] Datchik toka ACS712, 2020. URL: https://3d-diy.ru/wiki/arduino-datchiki/datchik-toka-acs712. (in Russian).
[11] S. Ryumik, Mikrokontrolleryi Wi-Fi, Seans 1, Radioamator 6 (2016) 42–43 (in Russian).
[12] Kak nauchit telefon varit tebe kofe i sohranyat kartinki bez ruk: obzor meshap-servisa IFTTT, 2020. URL: https://habr.com/ru/company/banderolka/blog/405617 (in Russian).
[13] S. Tsyrulnyk, V. Tkachuk, V. Roptanov, Prykladne prohramuvannia Embedded ta IOT prystroiv, in: Proceedings of the XIIth MNPK. Internet-Education-Science (IES-2020), Vinnytsia, 2020, pp. 243–245 (in Ukrainian).
[14] S. M. Tsyrulnyk, G. L. Lysenko, Design of microprocessor systems, Vinnytsia, VNTU, 2012, 191 p. (in Ukrainian).